CS 342

Operating Systems

Project 1

Kazım Ayberk Tecimer

21502531

Section-1

Part C:

A) Time measurement is done by both gettimeofday() method in C and also time systemcall in linux. These measurements include the all of the operations (file open, read, write, child, calculations) performed by parent/child processes and multi threads. Around 100-200 tests are done for a case to get more accurate result. Then, average and standard deviation of the experiments are calculated and written on the table and plotted.

Table 1

Running time of multi-process and multi-thread application for processes.

# of Processes/Threads	Processes elapsed time (µs)	Threads elapsed time (µs)
1	1526	991.7
2	2063	1247.4
4	3008	1793
8	5523	2332.9

Table 2

Standard Deviation of multi-process and multi-thread application

# of Processes/Threads	Standard Deviation in MultiProcessing	Standard Deviation in MultiThreading
1	120.5	81.9
2	214.6	108.3
4	241	176.2
8	324.9	217

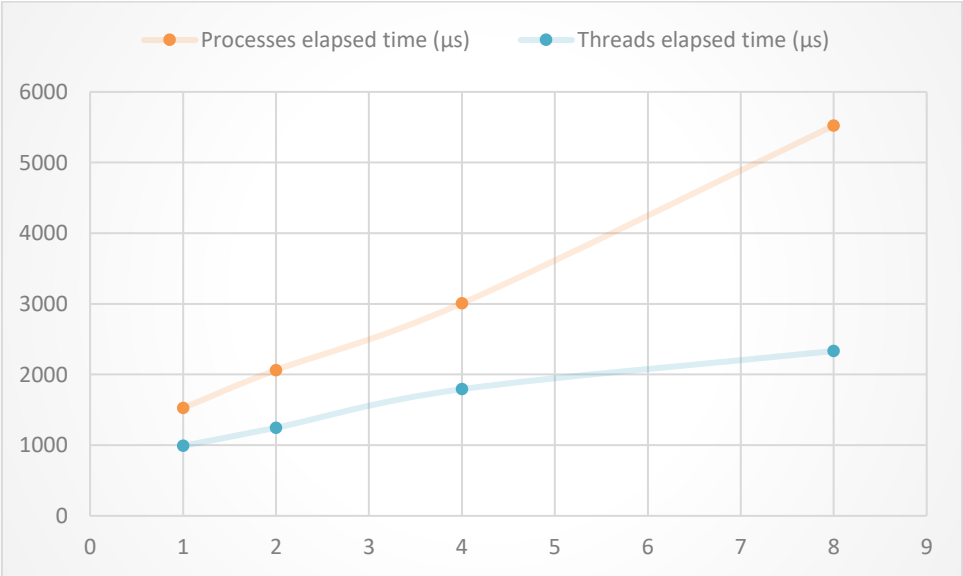

Figure 1. Relation between elapsed time and number of processes / threads.

From first experiment, it is observed that multiprocessing takes more time than the multhreading version. This is reasonable as multiprocessing is more heavy-weight than multithreading and reasons are as follows:

Threads share memory and resources yet every process run their its own address spaces.

Thread creation is fast whereas process creation expensive, and thread switching is lower overhead than context switching.

Lastly, in most of the problems multithreading is considerably faster than multiprocessing but when one thread failed in execution it also affects other threads and they may also fail executing but this is not case in multiprocessing.

B)

TABLE 3
Running time for 2 processes/threads for different input sizes.

File Size (# of Values	Processes elapsed time (µs)	Threads elapsed time (µs)
10	1028	771
100	1758.8	953.4
1000	2750	1587.2
10000	7663	2463

TABLE 4
Standard deviation of multi-process and mult-thread application with different file size

File Size (# of values)	Standard Deviation in MultiProcessing	Standard Deviation in MultiThreading
10	87.3	62.3
100	182.8	78.6
1000	249	164.7
10000	561	208

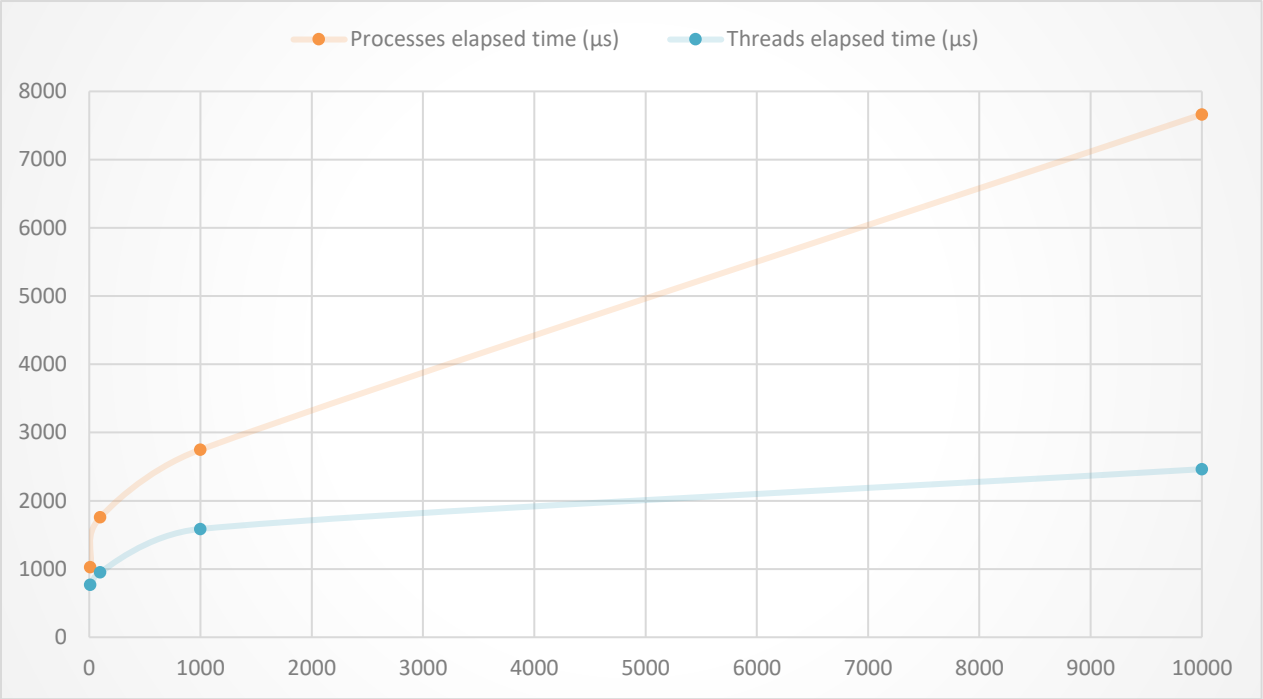

Figure 2. Relation between elapsed time and number of processes / threads.

In the second experiment, outcomes are in parallel with first experiment. It is observed that as input size increases, both multithreading and multiprocessing take much time, as expected, and much importantly as observed in the second experiment multithreading is faster than multiprocessing.